

CS 161: Section 6 Solutions

Midterm Recap

This was just general recap, so there are no solutions here.

1. General midterm review.

See midterm solutions and/or Piazza; in section we did a midterm Q&A.

2. How to do a formal proof by induction.

One of the things many people struggled with on the midterm was writing a formal proof by induction. Please take a look over your answer to problem 5, parts (c) and (d), at the TA's feedback, and at the solutions.

Here is a list of resources to help you out with proving things by induction:

CS103 Guide to Induction

<http://web.stanford.edu/class/cs103/handouts/240%20Guide%20to%20Induction.pdf>

CS103 Induction Checklist

<http://web.stanford.edu/class/cs103/handouts/300%20Induction%20Proofwriting%20Checklist.pdf>

Dijkstra's Algorithm

This was just general recap, so there are no solutions here.

In section we went through an example in detail; a great way to get this experience for yourself is to play around with this awesome website: <https://www.cs.usfca.edu/~galles/visualization/Dijkstra.html>

We also recapped the proof of Dijkstra's algorithm, so take a look at that in the course notes! It's a great example of a complex inductive argument.

Bonus Graph Problem: Farmville!

There are actually solutions in this section! See section handout for problem statements.

1. Let $G = (V, E)$ be the directed graph in which V is the set of farms, and there is a directed edge $(u, v) \in E$ for each pipe oriented from farm u to farm v .
2. To determine whether the condition holds:
 - Pick an arbitrary node $v \in V$.

- Run BFS in G from v and check that all of V is reachable from v ; if not, return “no”.
- Construct the reverse graph $G' = (V, E')$, where $E' = \{(v, u) : (u, v) \in E\}$.
- Run BFS in G' from v and check that all of V is reachable from v ; if not, return “no”. Otherwise return “yes”.

Running time: It takes $O(1)$ time to pick some node v , $O(m + n)$ time to run BFS on G , $O(m + n)$ to build G' from G (or just reverse the direction of existing edges in G), and $O(m + n)$ time to run BFS again on G' . This is $O(m + n)$ in total.

Correctness: Running BFS from v in G tells us whether v can reach all other nodes; running BFS from v in the reverse graph G' tells us whether all other nodes can reach v in G . This is clearly a necessary condition if we want all nodes to be able to reach one another; is it also sufficient? It is, since to get from any node u to any other node w , we can take the path $u \rightarrow \cdots \rightarrow v \rightarrow \cdots \rightarrow w$. Thus our algorithm outputs “yes” if and only if every node can reach every node, which corresponds to every farm being able to send water to every other farm.

3. (2 points) Algorithm: For each edge $e \in E$, run the algorithm of part (a) on $G \setminus \{e\}$, the graph obtained by removing e . Return “no” if any of these returns “no”; otherwise return “yes”.

Running time: We run the algorithm of part (a) m times, giving a running time of $O(m^2 + mn)$.

4. The minimum number of pipes/edges is $2n$. To see this, note that:

- It is necessary that for every node $v \in V$, $\text{indegree}(v) \geq 2$; that is, at least 2 edges pointing towards it. If not, then if we removed the sole edge (u, v) from G , then v cannot reach any other nodes. The same argument shows that we require $\text{outdegree}(v) \geq 2$. If this is the case, then the number of edges is

$$m = \frac{1}{2} \sum_{v \in V} \deg(v) \geq \frac{1}{2} \sum_{v \in V} 4 \geq 2n$$

where the $1/2$ comes from the fact that each edge contributes to the degrees of two nodes.

- The condition above is also sufficient. To see this, number the vertices $v_1, \dots, v_n$ and consider the “double-cycle” graph containing all edges (v_i, v_{i+1}) and (v_{i+1}, v_i) . This graph remains strongly connected if any single edge is removed, and it has exactly $2n$ edges.